

# Decoupling LabVIEW and Continuous Integration

Steve Ball, Composed Systems

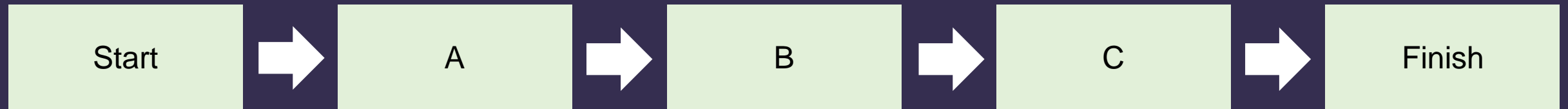
2019 CLA Americas 7x7

Why use CI/CD?

It's the year 2019,  
**DevOps** is a thing



CI/CD is **automation**



# What CI System should you use?



How does CI **connect** to LabVIEW?

**Execute batch command**

# Batch command arguments:

- LabVIEW Version
- LabVIEW Bitness
- LabVIEW Project file(s)
- What action/actions to perform
- How to perform those actions
- What working directory



By default, the CI Server must **know a lot** about your LabVIEW project

This is a **bad thing**\*

**A LabVIEW Project should  
know how to **test** and **build**  
itself**

# Introducing **Composed-CI**

(@cs/ci on gpackage.io)



**Configure and Test your LabVIEW-specific  
CI steps on your development machine**

**Reduce the complexity  
of your CI system**

**Reduce the number of commands and  
parameters  
in your test-and-build script**

**Reduce the setup and debug time  
when troubleshooting**

# Sweet UI

**Configure Build**

Config File

Open Save

Configure

Select Build Specs

Select Tests

Manual API

Open Project

Start Build

Run VI Tester

Run VI Analyzer

Packages

Relink Packages

Project

Recompile Project

Import Externals

The Hammer

Mass Compile

Working Directory

z:\labview\gpm\_packages\@cs\ci

GPM\_Packages directory

Z:\labview\gpm\_packages

Project File to Build

z:\labview\gpm\_packages\@cs\ci\composed-ci.lvproj

Build Specification Names

VI Analyzer configuration files

z:\labview\gpm\_packages\@cs\ci\Test\VIA\_tests\_gpm.cfg

VI Tester Directory

z:\labview\gpm\_packages\@cs\ci\Test

Run exclusively

Test composed-ciTest composed-ci.lvclass

Never Run

Check in the **JSON configuration file**  
with your project source code



# Capabilities

1. Run LabVIEW Build Specifications
2. Run VI Tester Tests
3. Run VI Analyzer
4. Install and Relink GPM Packages
5. Mass Compile (the built in way)
6. Recompile Project (slightly smarter way)
7. Detect Cyclic Dependencies

# Command Format

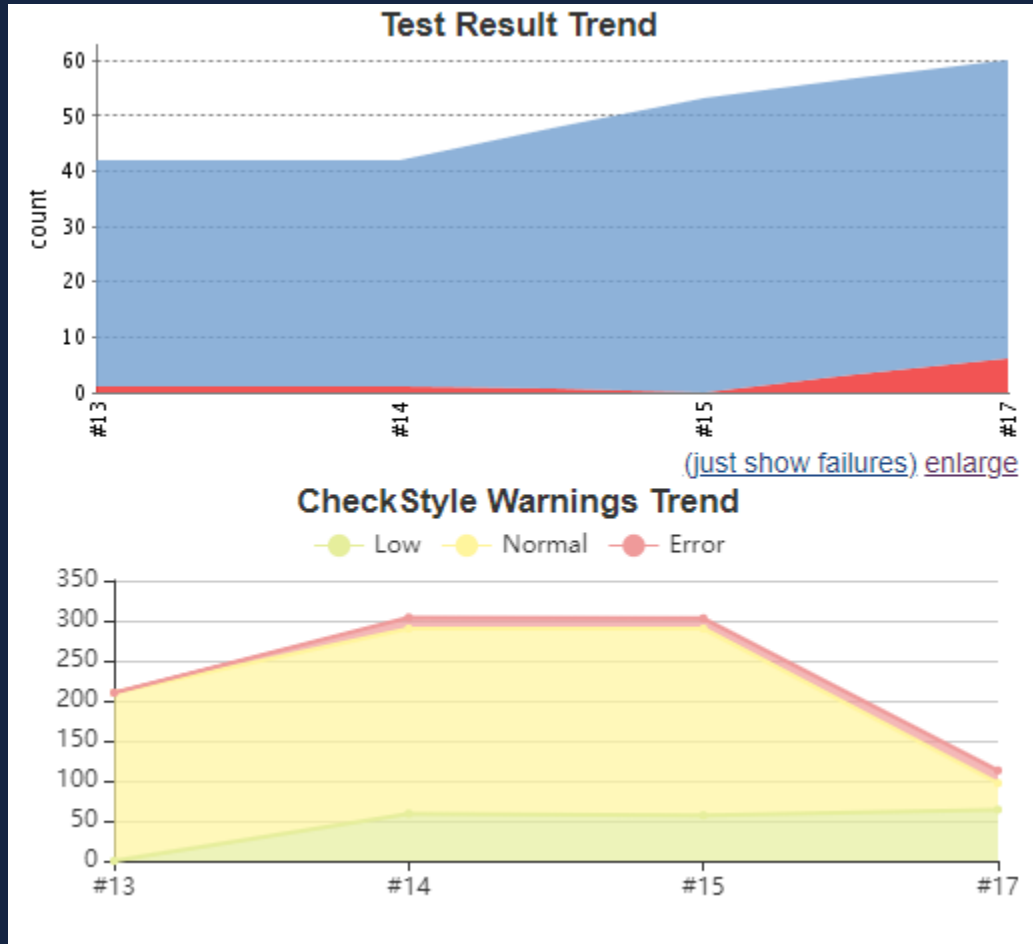
```
labview-cli --lv-ver %LV% "Call Build.vi" -- "-f composed-ci.json" "-d %WORKSPACE%"
```

1. LabVIEW Version
2. LabVIEW Bitness
3. What action to perform
4. Which configuration file
5. What working directory

# Useful Console Output

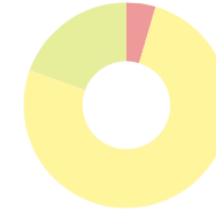
```
=====
Opening project at 8/22/2019 8:29 PM with the following arguments:
-f c:\jenkins\workspace\LabVIEW Modules\composed-ci\tools\composed-ci.cfg -d c:\jenkins\workspace\LabVIEW Modules\composed-ci
=====
=====
Finished opening project composed-ci.lvproj.
-found 72 valid VIs
=====
=====
Project Open Completed at 8/22/2019 8:29 PM
=====
```

# Reports!

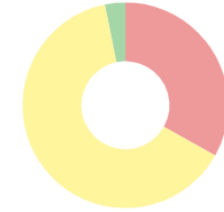


## CheckStyle Warnings

### Severities Distribution



### Reference Comparison



### Details

[Folders](#) [Files](#) [Issues](#)

Show 10 entries

Source Folder	Total
<a href="#">C:/jenkins/workspace/LabVIEW Modules/event logger/Examples</a>	46
<a href="#">C:/jenkins/workspace/LabVIEW Modules/event logger/Examples/GenericSerializable</a>	4
<a href="#">C:/jenkins/workspace/LabVIEW Modules/event logger/Source/Composed Log/Application Logger</a>	33
<a href="#">C:/jenkins/workspace/LabVIEW Modules/event logger/Source/Composed Log/DateFormat</a>	2

# Open Source

The screenshot shows a web browser with two tabs. The left tab is titled 'GPM: @cs/ci' and shows the 'gpackage.io/packages/@cs/ci' page. The right tab is titled 'composedsystems / composed-ci' and shows the Bitbucket source code for 'composed-ci' at 'bitbucket.org/composedsystems/composed-ci/src/master/'.

**GPM Registry Page (Left Tab):**

- Header: GPM Search the registry...
- Section: @cs/ci
- Sub-section: composed-ci
- Text: composed-ci takes the LabVIEW-specific configuration continuous integration.
- Text: This package exists because of the following premises:
  1. A LabVIEW project should know how to test and build itself.
  2. Configuration and testing of the LabVIEW-specific configuration should be able to be performed in the LabVIEW IDE. By managing LabVIEW's test-and-build configuration in the project, the API and number parameters the CI server needs to manage are minimized, which in turn decouples our projects from the CI server.
- Text: By managing LabVIEW's test-and-build configuration in the project, the API and number parameters the CI server needs to manage are minimized, which in turn decouples our projects from the CI server. This assists with managing responsibilities, and minimizes the amount of information the CI server has to know about your project.

**Bitbucket Source Code Page (Right Tab):**

- Header: Composed Systems / LabVIEW Modules
- Section: composed-ci
- Text: composed-ci is an extension of the LabVIEW CLI common steps package that takes the LabVIEW-specific configuration complexities out of your continuous integration. This package exists because of the following premises: 1. A LabVIEW project should know how to test and build itself. 2. Configuration and testing of the test and build steps should be performed in the LabVIEW IDE. By managing LabVIEW's test-and-build configuration in the project, the API and number parameters the CI server needs to manage are minimized, which in turn decouples our projects from the CI server.
- Navigation: master (dropdown), Filter files
- Table of files:

Name	Size	Last commit	Message
Source		2 days ago	- updated config file (new json forma...
Test		2 days ago	- updated config file (new json forma...
Tools		2 days ago	- updated config file (new json forma...
.gitignore	37 B	2019-02-28	0.9.17 - added tests for composed-ci....

# Thank You



**Steve Ball**

Steve.Ball@Composed.io

[www.composedsystems.com](http://www.composedsystems.com)

[bitbucket.org/composedsystems](https://bitbucket.org/composedsystems)

Find our packages on [gpackage.io](http://gpackage.io)